STATEMENT

I, Kazuhito Ino, a citizen of Japan, residing at Gifu-shi, Japan, hereby state that I believe the attached document is an accurate English translation of Japanese Patent Application No. 2002-286475, filed on September 30, 2002.

Kazuhito Ino

November 23, 2005

[Title of Document] Petition for Patent

[Reference Number] 0240491

[Filing Date] September 30, 2002

[To] The Commissioner of the Patent Office

[International Patent Classification] H04L 29/10

[Title of The Invention] INTERFACE CONVERTER AND INTERFACE CONVERTING METHOD

[Number of Claims] 10

[Inventor]

[Address] c/o FUJITSU VLSI LIMITED, 1844-2, Kozoji-cho 2-chome, Kasugai-shi, Aichi-ken

[Name] Manabu NAKANO

[Applicant for Patent Application]

[Applicant ID Number]000005223

[Name] FUJITSU LIMITED

[Agent]

[ID Number] 100068755

[Patent Attorney]

[Name] Hironori ONDA

[Appointed Agent]

[ID Number] 100105957

[Patent Attorney]

[Name] Makoto ONDA

[Amount of Charge]

[Receipt Number for Advance Payment] 002956

[Amount of Payment] 21,000 Yen

[List of Documents Attached]

[Name of Document] Specification 1

[Name of Document] Drawings 1

[Name of Document] Abstract 1

[Number of General Power of Attorney] 9909792

[Official Confirmation Required/Not Required] Required

[Title of Document] Specification

[Title of the Invention] INTERFACE CONVERTER AND INTERFACE CONVERTING METHOD

[Scope of the Invention]

[Claim 1] An interface converter characterized by:

first command converting means for converting a command between a first interface and a second interface;

second command converting means for converting a command between the first interface and a third interface; and

checking/switching means that checks whether the interface of the connected device complies with the second interface or the third interface and selects the first command converting means or the second command converting means in accordance with the interface of the device based on the checking result.

[Claim 2] The interface converter according to claim 1, characterized in that the checking/switching means issues a command to the connected device and checks the interface of the device based on status information generated by the device.

[Claim 3] The interface converter according to claim 1 or 2, characterized in that at least one of the first and second command converting means includes determining means for determining whether to convert a command received via the first interface to a command complying with the corresponding second interface or the third interface, the at least one of the first and second command converting means outputting an error status via the first interface when not converting the command based on the determination result.

[Claim 4] The interface converter according to any one of

claims 1 to 3, characterized by:

first status converting means for converting a status between the first interface and the second interface; and

second status converting means for converting a status between the first interface and the third interface,

wherein the checking/switching means selects the first status converting means or the second status converting means in accordance with the interface of the device based on the checking result.

[Claim 5] The interface converter according to any one of claims 1 to 4, characterized by:

first data converting means for converting a datum between the first interface and the second interface; and

second data converting means for converting a datum between the first interface and the third interface,

wherein the checking/switching means selects the first status converting means or the second status converting means in accordance with the interface of the device based on the checking result.

[Claim 6] The interface converter according to claim 5, characterized in that at least one of the first and second data converting means encodes or decodes input data to convert data.

[Claim 7] The interface converter according to claim 5, characterized in that at least one of the first and second data converting means adds an error correction code to the input data when converting data and outputs the converted data.

[Claim 8] The interface converter according to claim 5, characterized in that at least one of the first and second

data converting means determines whether input data includes a predetermined data pattern when converting data and stops outputting the input data when the predetermined data pattern is included.

[Claim 9] An interface converting method characterized by:

first command converting means for converting a command between a first interface and a second interface; and

second command converting means for converting a command between the first interface and a third interface,

wherein the method includes checking whether the interface of the connected device complies with the second interface or the third interface and selecting the first command converting means or the second command converting means in accordance with the interface of the device based on the checking result.

[Claim 10] The interface converting method according to claim 9, characterized by: sequentially issuing to the connected device a command complying with the second interface and a command complying with the third interface; and checking interface of the device in accordance with status information of the device for each of the commands.

[Detailed Description of the Invention]

[0001]

[Industrial Field of Application]

The present invention relates to an interface converter and an interface converting method.

[0002]

Interface standards progress at high speeds. Conventional devices have always been required to adapt to new interface standards. Therefore, both main devices and peripheral devices must be provided with an interface that complies with the new interface standards. For example, when using a peripheral device that complies with a new interface standard, an interface that complies with the new interface standard must be provided for a main device. However, it takes much time to develop a main device that has a new interface. Thus, there is a demand for an interface converter that adapts a main device to a new interface standard without newly developing a main device.

[0003]

[Prior Art]

A conventional main device, such as a personal computer, and peripheral devices are provided with an interface complying with the AT Attachment Packet Interface (ATAPI) standard or the AT Attachment (ATA) standard to perform data communication between the main device and the peripheral devices (refer to patent document 1).

[0004]

In recent years, interfaces complying with the Universal Serial Bus 2.0 standard are often used in main devices and peripheral devices to facilitate connection and disconnection of the peripheral devices.

In recent years, main devices and peripheral devices equipped with interfaces complying with the Universal Serial Bus 2.0 standard (hereafter simply referred to as USB) are becoming available on the market. For example, the USB

interface enables two devices to be connected and disconnected in a state in which the power supply is activated. Devices having USB have therefore become popular.

[0005]

Much time is required to newly develop a main device or a peripheral device provided with a USB interface. Thus, an interface converter is used to convert an interface complying with ATAPI or ATA standard to an interface complying with the USB standard.

[0006]

The interface converter converts commands, status, and data complying with the USB standard to those complying with one of the interfaces. That is, a single interface converter converts commands, status, and data complying with the USB standard to those complying with the ATAPI standard. Also, another interface converter converts commands, status, and data complying with the USB standard to those complying with the ATA standard.

[0007]

The employment of the interface converter enables a main device or a peripheral device complying with the USB standard to be developed within a short period of time.

[0008]

[Patent Document 1]

US Patent No. 5,715,274

[0009]

[Problems that the Invention is to Solve]

However, the connectors of the ATAPI and ATA connectors have identical shapes. Therefore, an incompliant interface

device could be connected to a connector. Therefore, for example, the interface converter that converts commands, status, and data complying with the USB standard to those complying with the ATAPI standard could be connected to a connector of an ATA device. However, the commands and layout of signal lines differ between an interface complying with the ATAPI standard and an interface complying with the ATA standard. Accordingly, there is a shortcoming in that communication cannot be performed with the USB device even though the interface converter is connected.

[0010]

Accordingly, it is an objective of the present invention to provide an interface converter that permits devices connected to the interface converter to identify each other and become accessible to each other, and to provide an interface converting method.

[0011]

[Means for Solving the Problems]

To achieve the above objective, the invention as set forth in claim 1 includes first command converting means, second command converting means, and checking/switching means. The first command converting means converts a command between a first interface and a second interface. The second command converting means converts a command between the first interface and a third interface. The checking/switching means checks whether the interface of the connected device complies with the second interface or the third interface and selects the first command converting means or the second command converting means in accordance with the interface of

the device based on the checking result. Therefore, the device complying with the first interface and the device connected to it identify each other and become accessible to each other.

[0012]

According to the invention as set forth in claim 2, the checking/switching means issues a command to the connected device and checks the interface of the device based on status information generated by the device. Therefore, the connected device is easily checked.

[0013]

According to the invention as set forth in claim 3, at least one of the first and second command converting means includes determining means for determining whether to convert a command received via the first interface to a command complying with the corresponding second interface or the third interface. The at least one of the first and second command converting means outputs an error status via the first interface when not converting the command based on the determination result. Therefore, the response to a command that does not comply with the second or third interface is fast.

[0014]

The invention as set forth in claim 4 includes first status converting means and second status converting means. The first status converting means converts a status between the first interface and the second interface. The second status converting means converts a status between the first interface and the third interface. The checking/switching

means selects the first status converting means or the second status converting means in accordance with the interface of the device based on the checking result.

[0015]

The invention as set forth in claim 5 includes first data converting means and second data converting means. The first data converting means converts a datum between the first interface and the second interface. The second data converting means converts a datum between the first interface and the third interface. The checking/switching means selects the first status converting means or the second status converting means in accordance with the interface of the device based on the checking result.

[0016]

According to the invention as set forth in claim 6, at least one of the first and second data converting means encodes or decodes input data to convert data. Therefore, even if a third person obtains the device to which the encoded data is recorded, the leakage of data is prevented.

[0017]

According to the invention as set forth in claim 7, at least one of the first and second data converting means adds an error correction code to the input data when converting data and outputs the converted data. Therefore, the reliability of the data is increased.

[0018]

According to the invention as set forth in claim 8, at least one of the first and second data converting means determines whether input data includes a predetermined data pattern when converting data and stops outputting the input

data when the predetermined data pattern is included. For example, if the data pattern is of computer viruses, contamination by the viruses is prevented. This improves the safety of the data.

[0019]

The invention as set forth in claim 9 includes first command converting means and second command converting means. The first command converting means converts a command between a first interface and a second interface. The second command converting means converts a command between the first interface and a third interface. The method includes checking whether the interface of the connected device complies with the second interface or the third interface and selecting the first command converting means or the second command converting means in accordance with the interface of the device based on the checking result. Therefore, the device complying with the first interface and the device connected to it identify each other and become accessible to each other.

[0020]

The invention as set forth in claim 10 sequentially issues to the connected device a command complying with the second interface and a command complying with the third interface and checks the type of the interface of the device in accordance with status information of the device for each of the commands. Therefore, the connected device is easily checked.

[0021]

[Embodiment]

One embodiment according to the present invention will now be described with reference to Figs. 1 to 10.

Fig. 1 is a block circuit diagram illustrating an interface converter 1.

[0022]

The interface converter 1 is connected to a first device (host device) 2, which has an interface complying with a first interface, or the Universal Serial Bus 2.0 (hereinafter, simply referred to as the USB). The interface converter 1 connects the first device 2 to a second device 3 (peripheral device), which has an interface complying with either one of a second interface, or the ATAPI, and a third interface, or the ATA. That is, the interface converter 1 is connected between the first device 2 and the second device 3. The interface converter 1 converts commands, status, and data between the USB and ATAPI and between the USB and ATA.

[0023]

The interface converter 1 includes a first control circuit 11, a second control circuit 12, a command converting circuit 13, a status converting circuit 14, a data converting circuit 15, and an interface checking/switching device (hereafter referred to as switching device) 16, which are checking/switching means in this embodiment.

[0024]

The first control circuit 11, which complies with the USB interface, converts electric signals processed in the interface converter 1 to electric signals complying with the USB standard and outputs the converted electric signals when transmitting signals. Further, the first control circuit 11

converts electric signals complying with the USB standard, to electric signals that are processed in the interface converter 1 when receiving signals. The second control circuit 12, which complies with the ATAPI and ATA interfaces, converts electric signals processed in the interface converter 1 to electric signals complying with the ATAPI standard or the ATA standard when transmitting signals. Further, the second control circuit 12 converts electric signals complying with the ATAPI or ATA standard, to electric signals that are processed in the interface converter 1 when receiving signals.
[0025]

The command converting circuit 13 includes a first command converter 21, which serves as a first command converting means, a second command converter 22, which serves as a second command converting means, and a switch circuit 23. The first command converter 21 converts a USB command to an ATAPI command and an ATAPI command to a USB command. The second command converter 22 converts a USB command to an ATA command and an ATA command to a USB command. The switch circuit 23 connects the first command converter 21 or the second command converter 22 to the first and second control circuits 11 and 12 in response to a first switch signal S1.
[0026]

The status converting circuit 14 includes a first status converter 24, which serves as a first status converting means, a second status converter 25, which serves as a second status converting means, and a switch circuit 26. The first status converter 24 converts a USB status to an ATAPI status and an ATAPI status to a USB status. The second status converter 25

converts a USB status to an ATA status and an ATA status to a USB status. The switch circuit 26 connects the first status converter 24 or the second status converter 25 to the first and second control circuits 11 and 12 in response to a second switch signal S2.

[0027]

The data converting circuit 15 includes a first data converter 27, which serves as a first data converting means, a second data converter 28, which serves as a second data converting means, and a switch circuit 29. The first data converter 27 converts USB data to ATAPI data and ATAPI data to USB data. The second data converter 28 converts USB data to ATA data and ATA data to USB data. The switch circuit 29 connects the first data converter 27 or the second data converter 28 to the first and second control circuits 11 and 12 in response to a third switch signal S3.

[0028]

The switching device 16 checks the interface type of the second device 3 connected to the interface converter 1 and generates the first to third switch signals S1 to S3 that are in accordance with the interface type. More specifically, the switching device 16 sends a command to the second device 3 via the second control circuit 12 and checks the interface type of the second device 3 based on a response from the second device 3. Further, the switching device 16 generates the switch signals S1 to S3 to connect the converters corresponding to the interface of the second device 3 to the second control circuit 12.

[0029]

In this manner, the interface converter 1 of the preferred embodiment includes the first and second command converters 21 and 22, the first and second status converters 24 and 25, and the first and second data converters 27 and 28 that respectively comply with the ATA and ATAPI standards. Further, the switching device 16 checks the interface type, or attribute, of the second device 3 to activate one of the command converters 21 and 22, one of the status converters 24 and 25, and one of the data converters 27 and 28 in accordance with the interface type. Since the converters optimal for the second device 3 is automatically selected, the first and second devices 2 and 3 are reliably connected to each other.
[0030]

The first and second command converters 21 and 22 will now be described.

Fig. 2 is a block circuit diagram showing the first command converter 21.

The first command converter 21 includes a plurality of (four in Fig. 2) memories 31, 32, 33, and 34, a selection circuit 35, a determination circuit 36, which serves as a determining means, and a conversion circuit 37. The memories 31 to 34 are sequentially provided with commands from the first device 2 via the USB.
[0031]

In accordance with the commands stored in the memories 31 to 34, the selection circuit 35 selects one of the memories 31 to 34 and provides the determination circuit 36 with the command stored in the selected memory. For example, the selection circuit 35 determines the order of selecting the

commands stored in the memories 31 to 34 in accordance with the order by which the second device 3 executes the commands. Depending on the order by which the second device 3 executes the commands, the total execution time of the commands could be reduced. Therefore, the selection circuit 35 selects the commands stored in the memories 31 to 34 so that the total execution time of the commands by the second device 3 is reduced.

[0032]

The determination circuit 36 determines whether or not to convert each of the provided commands. When a command must be converted, the determination circuit 36 provides the command to the conversion circuit 37. When a command does not have to be converted, the determination circuit 36 sends an error response output signal to the corresponding status converting circuit 14.

[0033]

More specifically, the determination circuit 36 receives a command list that lists the commands that generate an error response from a table (not shown). The determination circuit 36 refers to the command list and checks whether the commands received from the memories 31 to 34 match any of the commands in the command list. The determination circuit 36 provides the conversion circuit 37 with the received command when the received command does not match any of the commands in the command list. When the received command matches a command in the command list, the determination circuit 36 provides the error response output signal SER to the status converting circuit 14.

[0034]

In the status converting circuit 14, the first status converter 24 provides the first device 2 with an error status signal in response to the error response output signal SER. That is, when the first device 2 provides the interface converter 1 with a command that is not supported by the second device 3, the interface converter 1 checks the command and provides the first device 2 with the error status signal, which is in accordance with the checked command. In this case, the command is not converted in the interface converter 1 and the command does not reach the second device 3. This reduces the response time and improves response.

[0035]

The conversion circuit 37 performs only format conversion. A USB command is data consisting of a plurality of bytes and includes an ATA command (same operation being represented by the same code). Accordingly, the conversion circuit 37 of the first command converter 21 retrieves the ATA command from the USB command and outputs the retrieved command.

[0036]

The second command converter 22 is configured in the same manner as the first command converter 21 except in that the conversion circuit 37 functions differently. The conversion circuit 37 of the second command converter 22 includes a table 37a associating USB commands with ATAPI commands. Operations that are substantially the same are represented by different codes in the USB command and the ATAPI command. The conversion circuit 37 refers to the table 37a to convert a USB command to an ATAPI command and outputs the converted command.

[0037]

The first data converter 27 and the second data converter 28 will now be discussed.

Fig. 3 is a block diagram of the first data converter 27. The second data converter 28 is configured in the same manner as the first data converter 27 and will thus not be described.

[0038]

The first data converter 27 includes a data checker 41, an error correction code processor 42, an encoder 43, and a decoder 44.

The data checker 41 is provided with USB data and data pattern information. The data pattern information includes data patterns of computer viruses and is registered in a storage circuit, such as a memory. The data checker 41 checks whether the input data has a data pattern matching a data pattern included in the data pattern information. The data checker 41 stops outputting the data when it has a data pattern matching the data pattern information and outputs the input data to the error correction code processor 42 when it does not have a data pattern matching the data pattern information. The data checker 41 performs a virus check and outputs the input data when confirming that viruses are not included in the data. Thus, the data checker 41 prevents viruses from entering the second device 3.

[0039]

The error correction code processor 42 adds at least one of an error correction code (ECC) and an error detection code, such as a cyclic redundancy check (CRC), to the input data and then outputs the data. Further, the error correction code

processor 42 uses the error correction code or the error detection code included in the input data to detect and correct errors in the input data. Then, the error correction code processor 42 outputs the processed data. Accordingly, the error correction code processor 42 increases the reliability of the data transfer between the first device 2 and the second device 3.

[0040]

The encoder 43 encodes the input data in accordance with a predetermined algorithm and outputs the encoded data. The decoder 44 decodes the input data in accordance with an algorithm that is reversed from that of the encoder 43 and outputs the decoded data.

[0041]

The data encoded by the encoder 43 is output to the second device 3. When the second device 3 is a hard disk drive (HDD), data from the USB interface of the first device 2 is encoded and written to the HDD. The decoder 44 decodes the data read from the HDD and provides the first device 2 with the decoded data.

[0042]

In this case, when a third person obtains only the second device 3 (HDD), the third person would not be able to decode the encoded data recorded in the HDD. Thus, the third person would not be able to confirm the contents of the data recorded to the HDD. Accordingly, the interface converter 1 prevents the leakage of information.

[0043]

The operation of the switching device 16 will now be

discussed with reference to Fig. 4.

Fig. 4 is a flowchart showing the checking operation of the switching device 16.

The switching device 16 issues an identify device command for the second device 3 (step 51). The device identify command complies with the device having the ATA standard. In step S52, the switching device 16 reads status information of the second device 3, or data of a status register, and checks an error bit of the read data. When the error bit is "1", this indicates that the command is invalid (not supported). If the error bit is not "1" (NO), the switching device 16 determines that the second device 3 complies with the ATA standard (step S53).

[0044]

When the error bit is "1" in step S52, the switching device 16 issues an identify packet device command for the second device 3 (step S54). The device identify command complies with the device having the ATAPI standard. In step S55, the switching device 16 reads the data of the status register in the second device 3 and checks an error bit of the read data. When the error bit is "1", this indicates that the command is invalid (not supported). If the error bit is not "1" (NO), the switching device 16 determines that the second device 3 complies with the ATAPI standard (step S56).

[0045]

If the error bit is "1" in step S55, the switching device 16 proceeds to step S57 and determines that the second device 3 does not support the ATA and ATAPI standards and thus cannot be identified. Accordingly, the switching device 16

disconnects the converters 21, 22, 24, 25, 27, and 28 from the first and second control circuits 11 and 12.

[0046]

The operation of the first command converter 21 will now be discussed with reference to Figs. 5 and 6.

Referring to Fig. 5, a command block 61a complying with the USB standard has thirty-one bytes of data.

[0047]

The first command converter 21 refers to the first to fourth bytes of input data and the data size of the command block 61a to determine whether the input data is a command complying with the USB standard. The fifth to eighth bytes of the data configure a tag code. The first command converter 21 stores the tag code in its memory and uses the tag code in a status response. The ninth to twelfth bytes of the data indicate the transfer data size and are referred to when transferring data. The thirteenth byte of the data is a data transfer flag. The fourteenth byte of the data indicates a logic device number of the first device 2. The fifteenth byte of the data indicates a valid byte number of the following command.

[0048]

The first command converter 21 extracts the sixteenth byte to the twenty-seventh byte and generates extracted data 62a, which complies with the ATAPI standard. The extracted data 62a is provided to the second device 3. The first command converter 21 ignores the effective data number in the fifteenth byte. This is because the ATAPI command provided from the second device 3 matches the valid data portion of the

USB command block 61a.

[0049]

The determination circuit 36 (shown in Fig. 2) of the first command converter 21 determines whether the twelve bytes of the extracted data 62a is a command supported by the second device 3 (ATAPI). The sixteenth byte is "12h" (h representing a hexadecimal, Fig. 5 shows only "12") and is a command supported by the ATAPI standard. Thus, the conversion circuit 37 (shown in Fig. 2) of the first command converter 21 generates an ATAPI packet 63a from the extracted data 62a and outputs the ATAPI packet 63a.

[0050]

Fig. 6 shows a command block 61b complying with the USB standard in which the sixteenth to twenty-seventh bytes are extracted to generate extracted data 62b. The head of the extracted data 62b is "25h" and is a command that is not supported by the ATAPI standard. Accordingly, the determination circuit (Fig. 2) of the first command converter 21 determines that the extracted data 62b is a command that does not have to be converted and does not output the extracted data 62b. In this case, the first command converter 21 does not generate an ATAPI packet 63b. Further, the determination circuit 36 outputs the error response output signal SER to notify the first device 2 of an error.

[0051]

The operation of the second command converter 22 will now be discussed with reference to Figs. 7 and 8.

In the same manner as the first command converter 21, the second command converter 22 extracts the sixteenth to twenty-

seventh bytes of a command block 64a, which complies with the USB standard. The second command converter 22 determines whether or not to convert the command portion based on the data of the sixteenth byte.

[0052]

Fig. 7 shows the command block 64a in which the data of the sixteenth byte is "28h" and is a command supported by the ATA standard. Accordingly, the second command converter 22 refers to the table 37a (Fig. 2) to generate a command 66a, which complies with the ATA standard, from extracted data 65a.

[0053]

When the extracted data 65a is a command for performing data transfer, the second command converter 22 checks the quantity of the transferred data in the twenty-third and twenty-fourth bytes. When the quantity of the transfer data exceeds 256 sectors, the second command converter 22 divides the command 66a into a plurality of commands. In this state, the second command converter 22 generates a command 67a, the address of which is incremented. The command 67a complies with the ATA standard.

[0054]

Referring to Fig. 8, in a command block 64b complying with the USB standard, the data of the sixteenth byte is "12h", which is a command that is not supported by the ATA standard. Accordingly, the second command converter 22 does not convert the extracted data 65b, that is, the determination circuit does not provide the conversion circuit with the extracted data 65b. As a result, the second command converter 22 does not generate commands 66b and 67b, which comply with the ATA

standard. In the same manner as the determination circuit 36 of the first command converter 21, the determination circuit 36 of the second command converter 22 outputs the error response output signal SER to notify the first device 2 of an error.

[0055]

The operation of the first status converter 24 will now be discussed with Fig. 9.

The ATAPI interface of the second device 3 is set to assert an interrupt request signal (INTRQ signal). When an error occurs or a command ends, the ATAPI interface asserts the INTRQ signal. Whenever INTRQ is asserted, the first status converter 24 reads the data of a status register 71 incorporated in the second device 3. The first status converter 24 generates a status block 72, which complies with the USB standard, based on the content of the status register 71.

[0056]

When the seventh bit of the status register 71 is "0", the status register 71 is valid. In this case, the first status converter 24 determines whether or not there is an error based on the zero bit (error bit). There is no error when the zero bit is "0", and there is an error when the zero bit is "1". The first status converter 24 sets the thirteenth byte of the status block 72 in accordance with the zero bit of the status register 71. The "0" of the zero bit is converted to "00h" and the "1" of the zero bit is converted to "01h".

[0057]

The first status converter 24 sets the remaining quantity

of the data transferred by the USB interface in the ninth to twelfth bytes of the USB status block 72. The tag code (e.g., the fifth to eighth bytes of the USB command block 72 shown in Fig. 5) held by the first command converter 21 is stored in the fifth to eighth bytes. The first status converter 24 stores an identification code indicating the status of the USB interface in the first to fourth bytes of the status block 72 and transfers the first to fourth bytes to the first device 2 (USB host).

[0058]

The operation of the second status converter 25 will now be discussed with reference to Fig. 10.

Whenever a command issued by the second command converter 22 ends, the second status converter 25 reads the data of a status register 73 in the second device 3. In this state, the second status converter 25 stores the quantity of the remaining sectors in the ninth to twelfth bytes of a status block 74 when an error occurs based on the content of zero bit of the status register 73. The second status converter 25 stores "01h" in the thirteenth byte of the status block 74. Further, the second status converter 25 stores the tag code held by the second command converter 22 in the fifth to eighth bytes of the status block 74. The second status converter 25 stores an identification code (signature) indicating the status of the USB interface in the first to fourth bytes of the status block 74 and transfers the status block 74 as an error response to the first device 2 (USB host).

[0059]

When all of the commands in a USB command end normally,

the second status converter 25 stores "00h" in the ninth to thirteenth bytes of the USB status block 74. The second status converter 25 stores the tag code held by the second command converter 22 in the fifth to eighth bytes, stores an identification code indicating a USB status in the first to fourth bytes, and performs a status response.

[0060]

The preferred embodiment has the advantages described below.

(1) The interface converter 1 includes the first command converter, which complies with the USB and ATAPI standards, and a second command converter 22, which complies with the USB and ATA standards. The switching device 16 checks the interface type of the connected second device 3 and connects the first command converter 21 or the second command converter 22 to the second device 3 via the control circuit 12 in accordance with the interface type. In accordance with the connected second device 3, the first command converter 21 and the second command converter 22 are switched. This ensures the connection of the second device 3 and the first device 2.

[0061]

(2) The determination circuit 36, which is provided in each of the first and second command converters 21 and 22, determines whether or not to convert a command from the USB interface to an ATAPI or ATA command and generates the error response output signal SER when determining not to convert the USB command. The first status converter 24 or the second status converter 25 provides the first device 2 with the error status signal in response to the error response output signal

SER. As a result, the interface converter 1 responds to commands that are not supported by the ATAPI or ATA standard. This improves response.

[0062]

(3) The data checker 41, which is provided in each of the first and second data converters 27 and 28, checks whether or not the input data has a portion that matches a predetermined data pattern and does not output the data when there is a matching portion. As a result, invalid data, such as a computer virus, is prevented from being transferred to the first and second devices 2 and 3.

[0063]

(4) The error correction code processor 42, which is provided in each of the first and second data converters 27 and 28, adds error correction and error detection codes to data and outputs the processed data. This prevents erroneous data from being transferred between the interfaces.

[0064]

(5) The encoder 43 and the decoder 44, which are provided in each of the first and second data converters 27 and 28, respectively encodes and decodes data. The second device 3, such as a HDD, records encoded data. Thus, even if a third person obtains the second device 3, the read data is encoded. This prevents data leakage.

[0065]

The preferred embodiment may be modified as follows.

In the preferred embodiment, the various interface converting processes are performed with the interface converter 1, or hardware. However, some of the functions of

the interface converter 1 may be performed with software.

[0066]

For example, the interface converter 81 of Fig. 11 includes a converting section 82 and an MPU 83, which are configured on a single chip. The converting section 82 includes first and second control circuits 11 and 12, a command converting circuit 13, a status converting circuit 14, and a data converting circuit 15 like the preferred embodiment. The MPU 83 executes a program 84, which functions as an interface checking/switching device. The program 84 is stored in a memory of the MPU 83 and in memories accessed by the MPU 83.

[0067]

Since the MPU 83 executes functions as the interface checking/switching device, checking sequences and issued commands may easily be changed. The functions as the interface checking/switching device are required to be executed only when the second device 3 is connected or when the interface converter 81 is activated. Thus, the program does not have to be executed constantly. Therefore, the functions as the interface checking/switching device are not achieved by the hardware. This reduces the power consumption of the interface converter 81 (converting section 82).

[0068]

Commands, status, and data are converted between the USB interface and the ATA or ATAPI interfaces. However, communication between the USB interface and the ATA or ATAPI interface is enabled as long as at least the command is converted. Accordingly, at least one of the status converting

circuit 14 and the data converting circuit 15 may be eliminated from the interface converter 1 of Fig. 1. As shown in Fig. 12, an interface converter 81a may be configured by a converting section 82a, which includes first and second control circuits 11 and 12, a command converting circuit 13, and a status converting circuit 14, and an MPU 83.

[0069]

The converting function and the checking function of the interface converter 1 may be performed with software.

For example, referring to Fig. 13, an interface converter 81b includes a converting section 82a and an MPU 83a, which are configured on a single chip. The converting section 82b includes first and second control circuits 11 and 12. The MPU 83a executes a checking program 84, which functions as an interface checking/switching device, a command conversion program 85, which functions as a command converting circuit, and a status conversion program 86, which functions as a status converting circuit. The MPU 83 may also be configured to execute a data conversion program, which functions as a data converting circuit. The programs 84, 85, and 86 are stored in a memory of the MPU 83a or in a memory accessed by the MPU 83a.

[0070]

The command conversion program 85 includes a program 91 functioning as a first command converter, a program 92 functioning as a second command converter, and a program 93 functioning as a switching circuit. The MPU 83a executes the program 91 to convert a command between a first interface and a second interface and executes the program 92 to convert a

command between the first interface and a third interface. Based on a command generated from the checking program 84, the program 93 switches the execution of the program 91 and the program 92 (for example, switches a pointer indicating the command executed by the MPU 83a).

[0071]

The status conversion program 86 includes a program 94 functioning as a first status converter, a program 95 functioning as a second status converter, and a program 96 functioning as a switching circuit. The MPU 83a executes the program 94 to convert a status between the first interface and the second interface and executes the program 95 to convert a status between the first interface and the third interface. Based on a command generated from the checking program 84, the program 96 switches the execution of the program 94 and the program 95 (for example, switches a pointer indicating the command executed by the MPU 83a).

[0072]

In such a configuration, even if the interfaces of the first device 2 or the second device 3 are changed, programs may be switched to perform conversions accordingly. This facilitates adaptation to a new interface within a short period of time.

[0073]

In each of the above embodiments, the conversion process for adapting to an ATA interface and the conversion process for adapting to an ATAPI interface are both performed with hardware or software. However, one of the conversion processes may be performed with hardware and the other one of

the conversion processes may be performed with software.

[0074]

The above mentioned embodiments are summarized as follows.

(Addition 1) An interface converter, characterized by:

first command converting means for converting a command between a first interface and a second interface;

second command converting means for converting a command between the first interface and a third interface; and

checking/switching means for checking whether the interface of the connected device complies with the second interface or the third interface and selects the first command converting means or the second command converting means in accordance with the interface of the device based on the checking result.(1)

(Addition 2) The interface converter according to addition 1, characterized in that the checking/switching means issues a command to the connected device and checks the interface of the device based on status information generated by the device.(2)

(Addition 3) The interface converter according to addition 1 or 2, characterized in that at least one of the first and second command converting means includes determining means for determining whether to convert a command received via the first interface to a command complying with the corresponding second interface or the third interface, the at least one of the first and second command converting means outputting an error status via the first interface when not converting the command based on the determination result.(3)

(Addition 4) The interface converter according to any one

of additions 1 to 3, characterized in that at least one of the first and second command converting means converts the command using a table that stores commands, which are associated with each other.

(Addition 5) The interface converter according to any one of additions 1 to 4, characterized by:

first status converting means for converting a status between the first interface and the second interface; and

second status converting means for converting a status between the first interface and the third interface,

wherein the checking/switching means selects the first status converting means or the second status converting means in accordance with the interface of the device based on the checking result.(4)

(Addition 6) The interface converter according to any one of additions 1 to 5, characterized by:

first data converting means for converting a datum between the first interface and the second interface; and

second data converting means for converting a datum between the first interface and the third interface,

wherein the checking/switching means selects the first status converting means or the second status converting means in accordance with the interface of the device based on the checking result.(5)

(Addition 7) The interface converter according to addition 6, characterized in that at least one of the first and a data converting means encodes or decodes input data to

convert data.(6)

[Claim 8] The interface converter according to addition 6, characterized in that at least one of the first and second data converting means adds an error correction code to the input data when converting data and outputs the converted data.(7)

(Addition 9) The interface converter according to addition 6, characterized in that at least one of the first and second data converting means determines whether input data includes a predetermined data pattern when converting data and stops outputting the input data when the predetermined data pattern is included. (8) An interface converting method characterized by:

(Addition 10) first command converting means for converting a command between a first interface and a second interface; and

second command converting means for converting a command between the first interface and a third interface,

wherein the method includes checking whether the interface of the connected device complies with the second interface or the third interface and selecting the first command converting means or the second command converting means in accordance with the interface of the device based on the checking result.(9)

(Addition 11) The interface converting method according to addition 10, characterized by: sequentially issuing to the connected device a command complying with the second interface and a command complying with the third interface; and checking

the type of the interface of the device in accordance with status information of the device for each of the commands.(10)

(Addition 12) The interface converting method according to addition 10 or 11, characterized in that at least one of the first and second command converting means includes determining means for determining whether to convert a command received via the first interface to a command complying with the corresponding second interface or the third interface, the at least one of the first and second command converting means outputting an error status via the first interface when not converting the command based on the determination result.

[0075]

[Effects of the Invention]

As described above, the present invention enables devices connected to each other to identify each other and become accessible to each other.

BRIEF DESCRIPTION OF THE DRAWINGS

[Fig. 1] A block circuit diagram of an interface converter according to one embodiment.

[Fig. 2] A block circuit diagram of a command converter.

[Fig. 3] A block circuit diagram of a data converter.

[Fig. 4] A flowchart illustrating an interface checking process.

[Fig. 5] A diagram illustrating the operation of a first command converter.

[Fig. 6] A diagram illustrating the operation of the first command converter.

[Fig. 7] A diagram illustrating the operation of a second command converter.

[Fig. 8] A diagram illustrating the operation of the second command converter.

[Fig. 9] A diagram illustrating the operation of a first status converter.

[Fig. 10] A diagram illustrating the operation of a second status converter.

[Fig. 11] A block circuit diagram of an interface converter according to a further embodiment.

[Fig. 12] A block circuit diagram of an interface converter according to a further embodiment.

[Fig. 13] A block circuit diagram of an interface converter according to a further embodiment.

[Description of the Reference Numerals]

1 interface converter

2 device (device having first interface)

3 device (device having second or third interface)

16 checking/switching means (interface checking/switching device)

21 first command converting means (first command converter)

22 second command converting means (second command converter)

24 first status converting means (first status converter)

25 second status converting means (second status converter)

27 first data converting means (first data converter)

28 second data converting means (second data converter)

[Title of Document] Abstract

[Abstract]

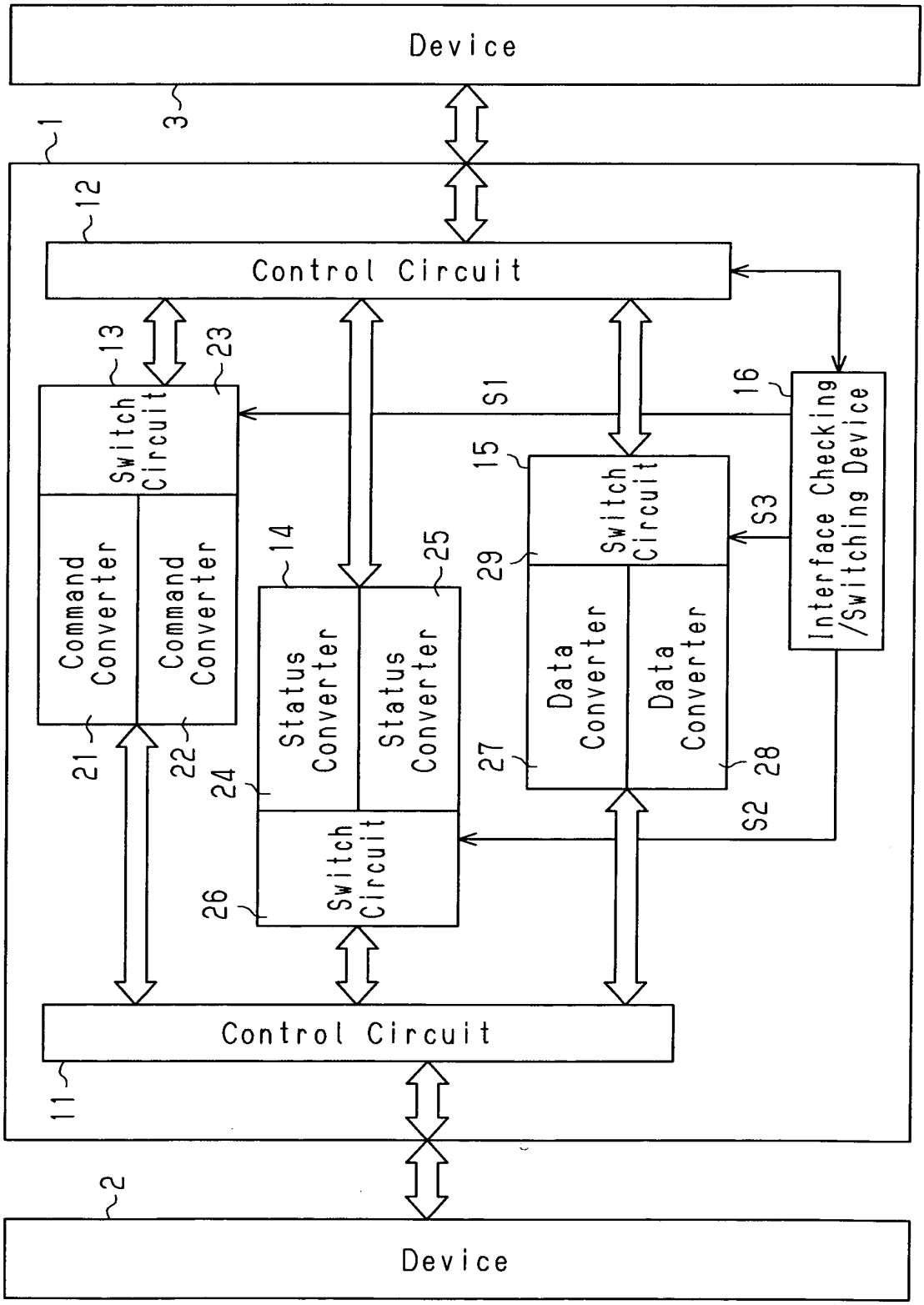
[Objective] To provide an interface converter that permits devices connected to each other to identify each other and become accessible to each other.

[Means for Solving the Problems] An interface converter 1 includes a first command converter 21, which complies with the USB and ATAPI standards, and a second command converter 22, which complies with the USB and ATA standards. A switching device 16 checks the interface type of a connected second device 3 and connects the first command converter 21 or the second command converter 22 to the second device 3 via a control circuit 12 in accordance with the interface type.

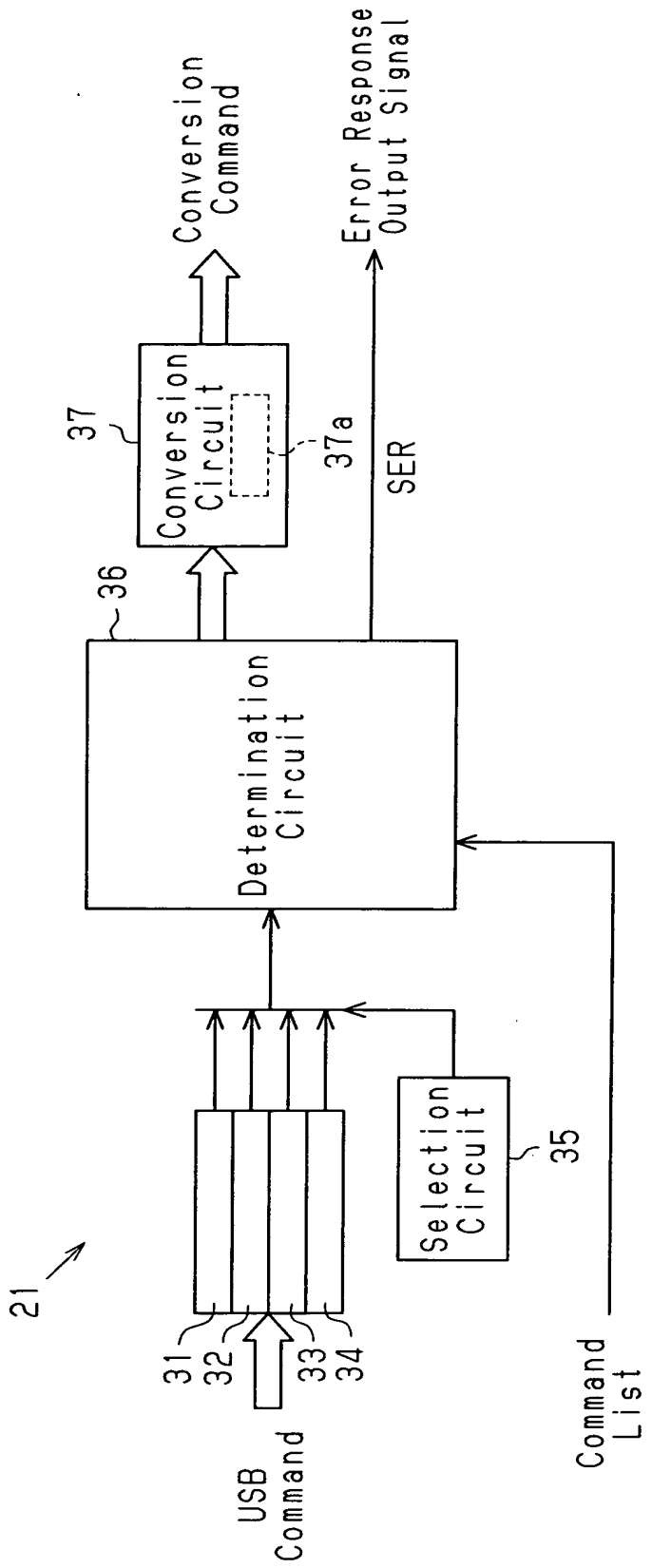
[Selected Drawing] Fig. 1

[Title of Document] Drawings

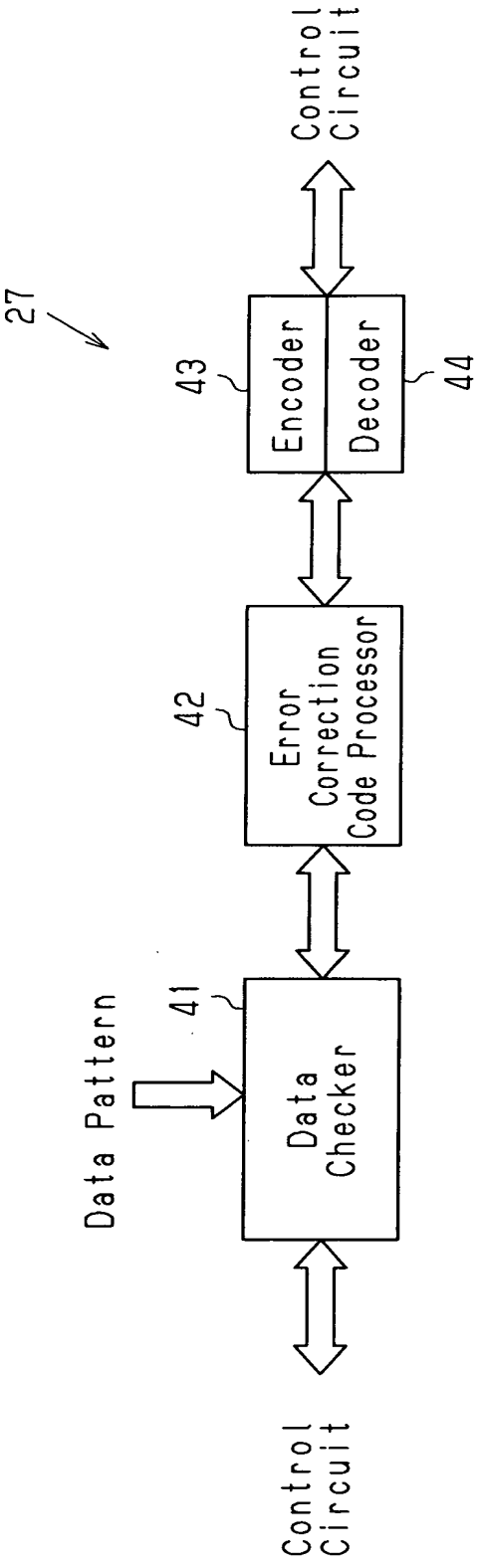
[Fig.1]



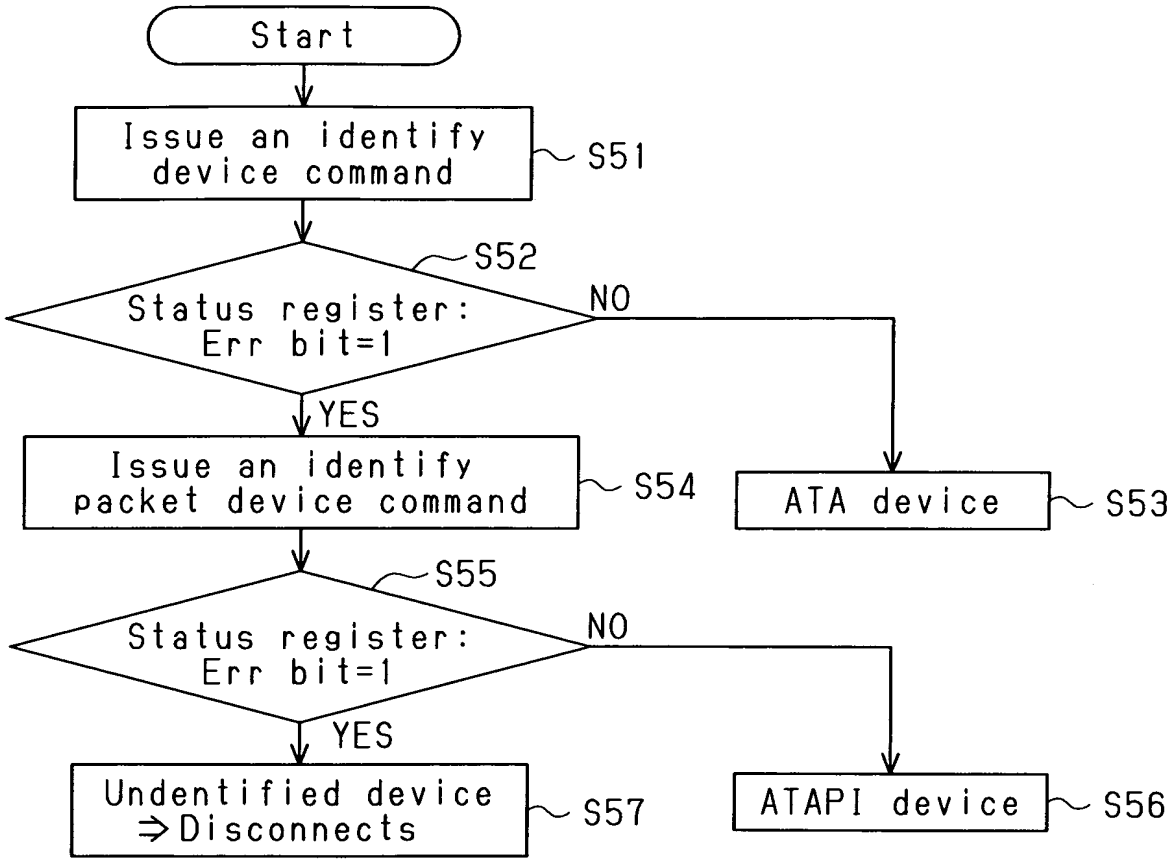
[Fig.2]


Conversion Command
Error Response Output Signal
37
Conversion Circuit
37a
SER
36
Determination Circuit
Selection Circuit
35
21
31
32
33
34
USB Command
Command List

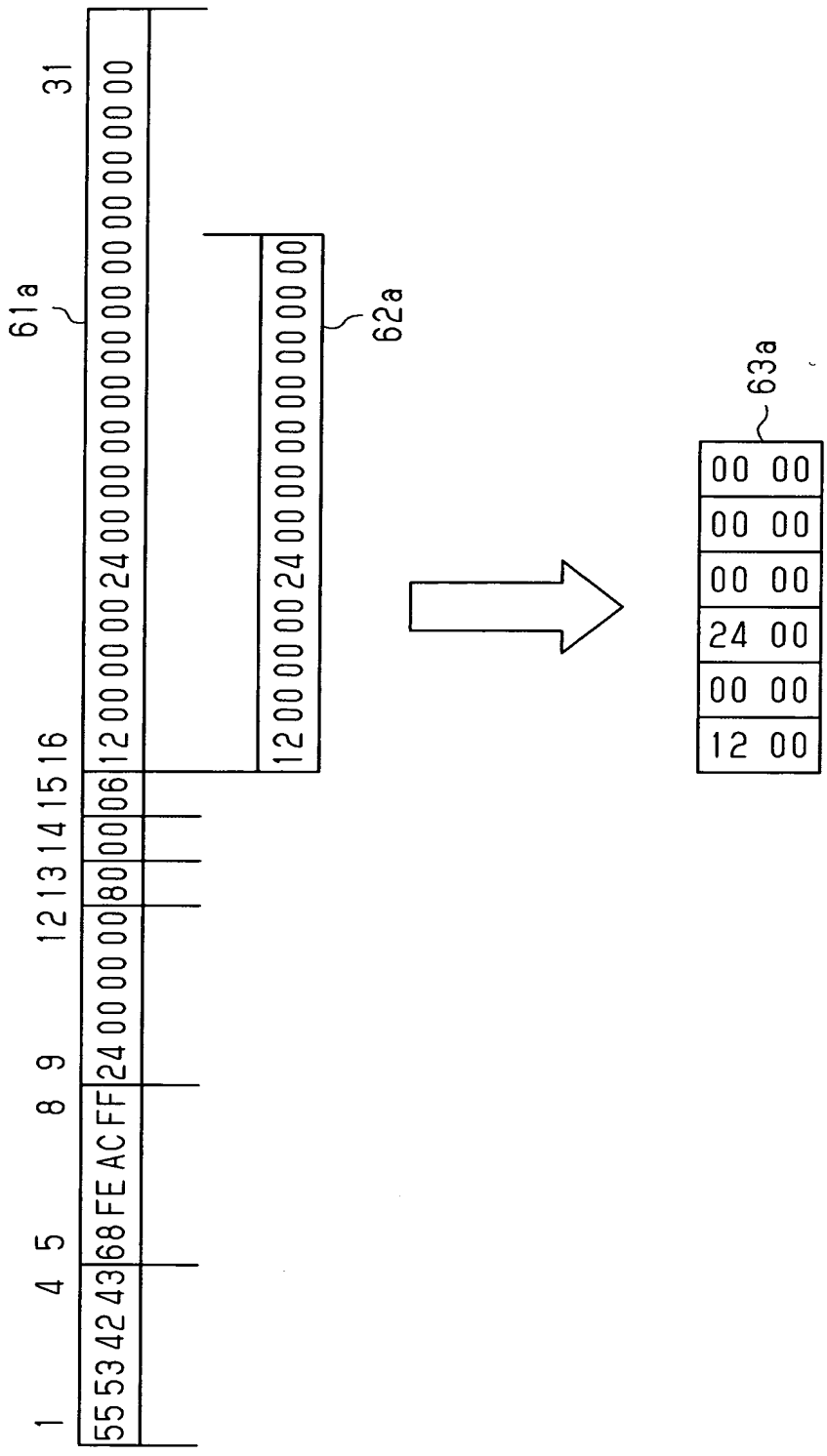
[Fig.3]


27
Control Circuit
43
Encoder
Decoder
44
42
Error Correction Code Processor
41
Data Pattern
Data Checker
Control Circuit

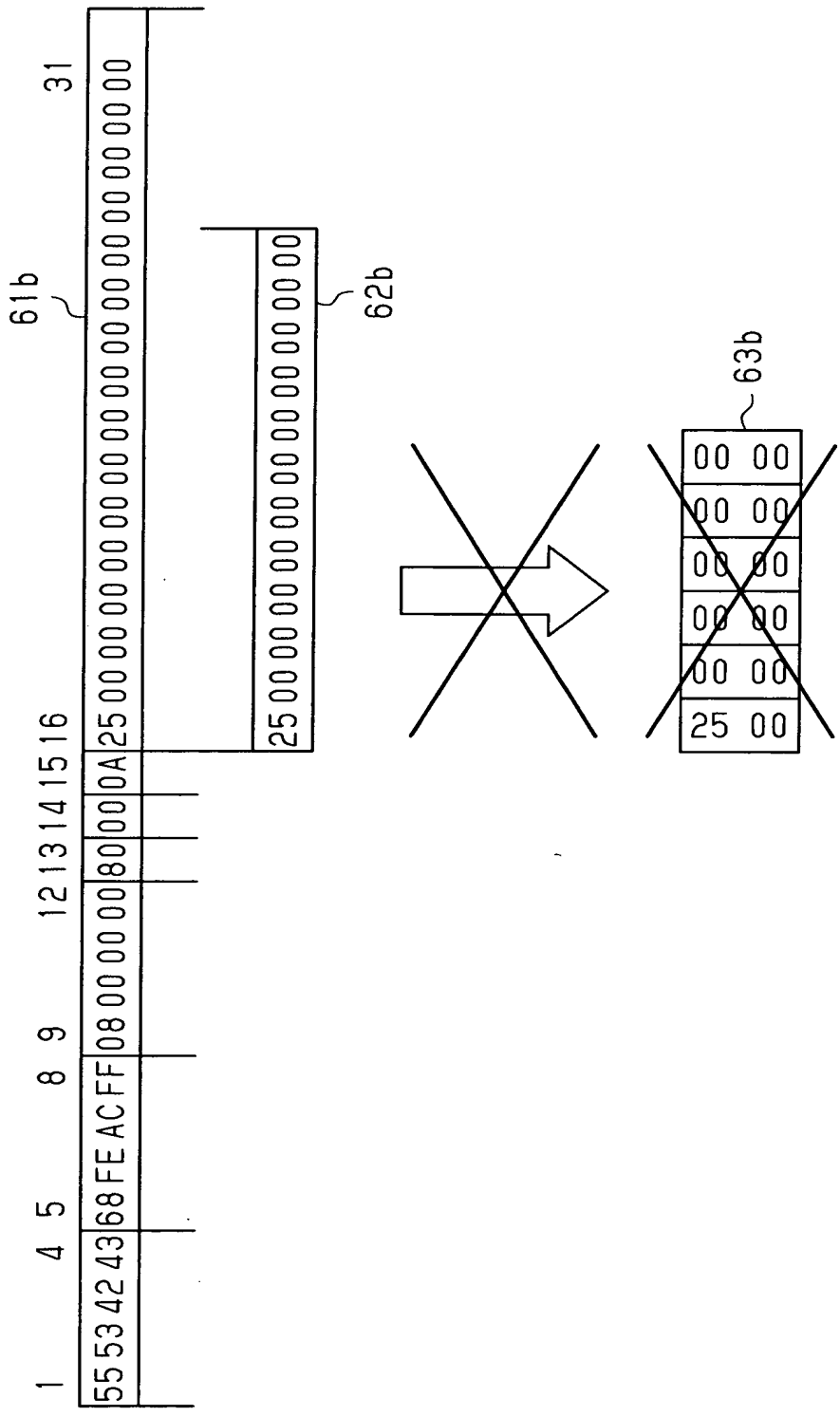
[Fig.4]


Start
Issue an identify device command
S51
S52
Status register: Err bit=1
NO
YES
Issue an identify packet device command
S54
ATA device
S53
S55
Status register: Err bit=1
NO
YES
Undentified device ⇒Disconnects
S57
ATAPI device
S56

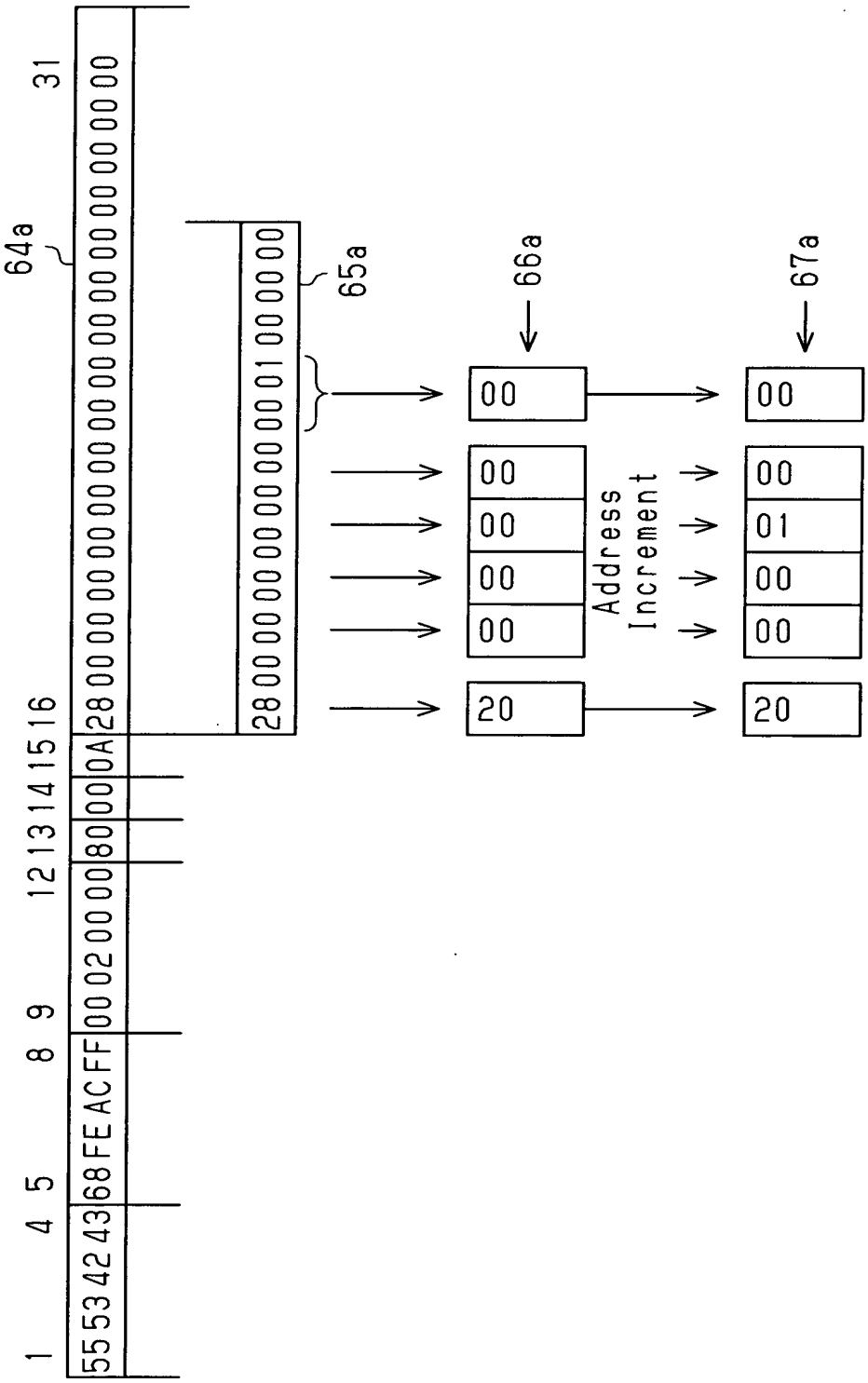
[Fig.5]


1
4 5
8 9
12 13 14 15 16
31
55 53 42 43 68 FE AC FF 24 00 00 00 80 00 06 12 00 00 00 24 00 00 00 00 00 00 00 00 00 00 00 00 00 00
61a
12 00 00 00 24 00 00 00 00 00 00 00
62a
63a
00 00
00 00
00 00
24 00
00 00
12 00

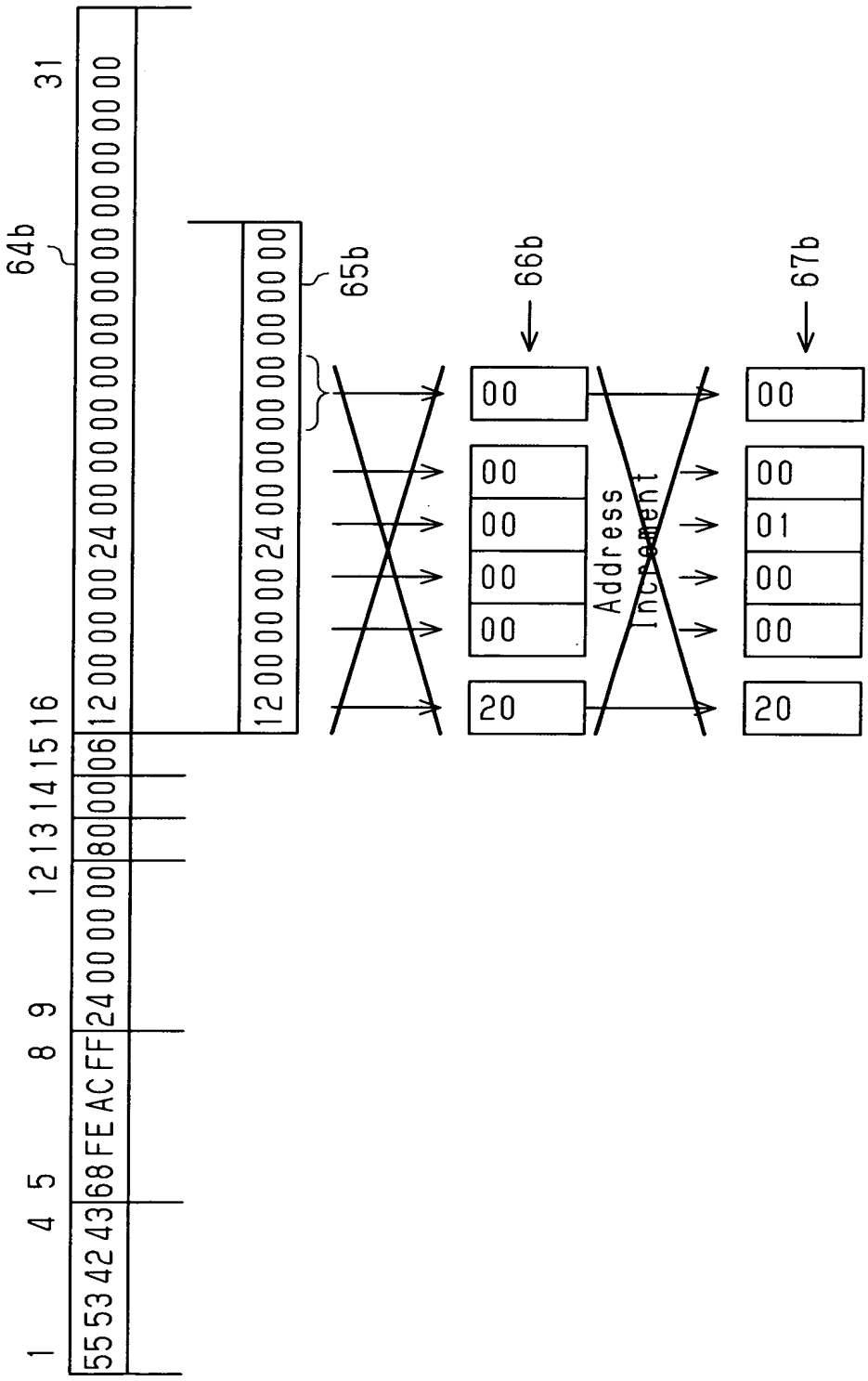
[Fig.6]


1
4 5
8 9
12 13 14 15 16
31
55 53 42 43 68 FE AC FF 08 00 00 00 80 00 0A 25 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
61b
62b
25 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
63b
00 00
00 00
00 00
00 00
00 00
25 00

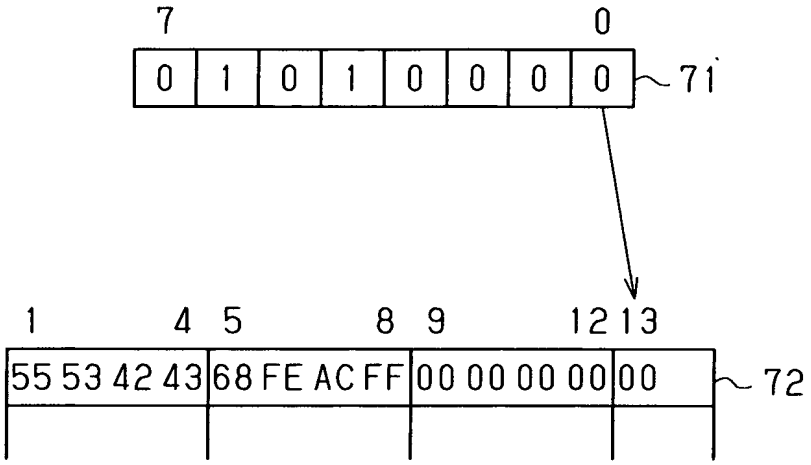
[Fig.7]


1
4 5
8 9
12 13 14 15 16
31
55 53 42 43 68 FE AC FF 00 02 00 00 80 00 0A 28 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
64a
28 00 00 00 00 00 00 00 00 01 00 00 00
65a
66a
67a
00
00
00
00
20
Address Increment
00
01
00
00
20

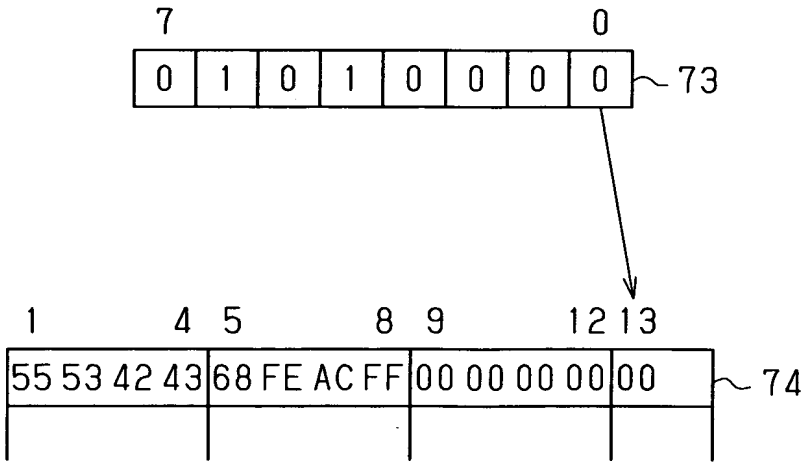
[Fig.8]


64b
65b
66b
67b
1 4 5 8 9 12 13 14 15 16 31
55 53 42 43 68 FE AC FF 24 00 00 00 80 00 06 12 00 00 00 24 00 00 00 00 00 00 00 00
12 00 00 00 24 00 00 00 00 00 00 00
00
00
00
00
20
Address Increment
00
00
01
00
00
20

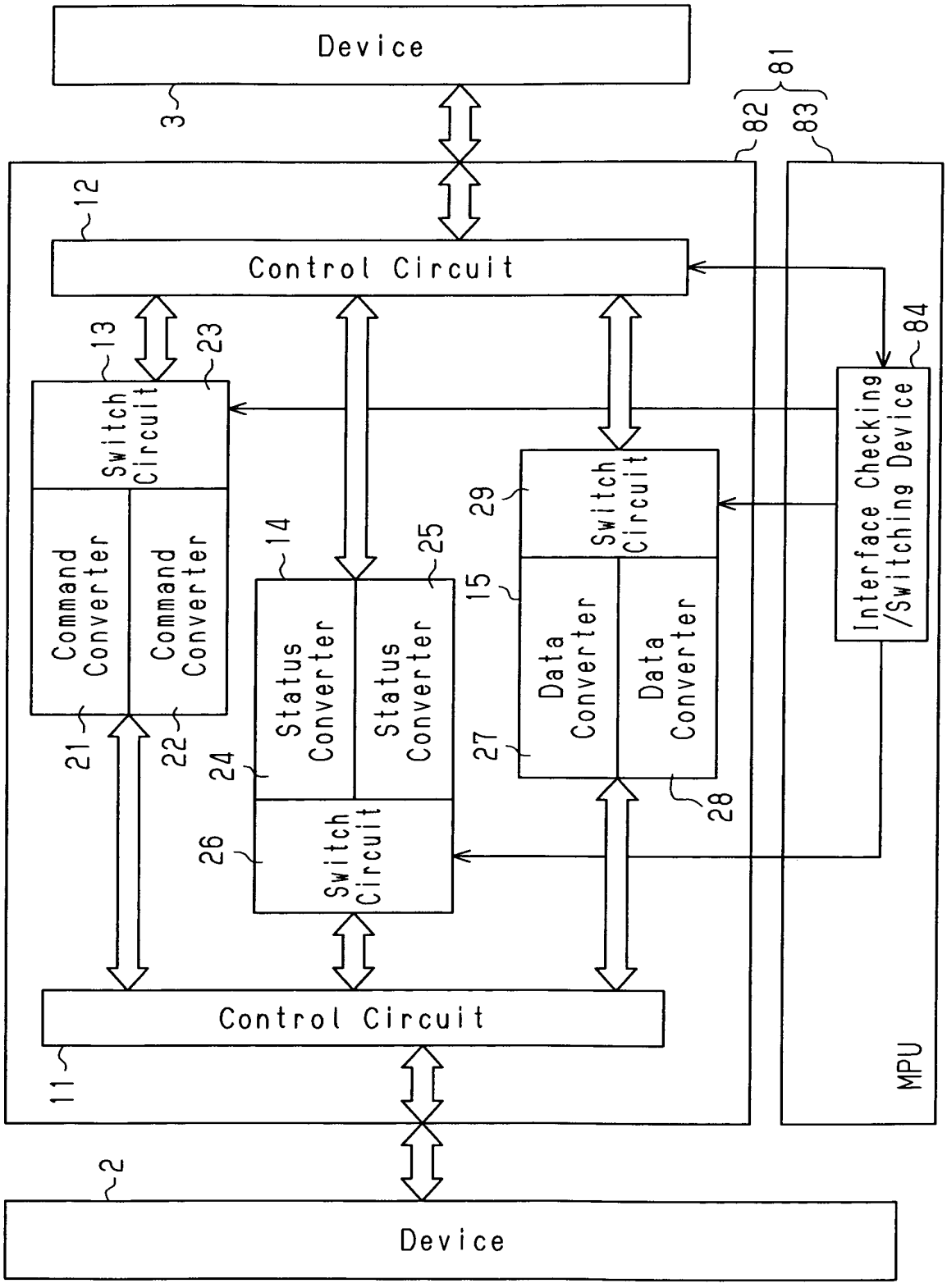
[Fig.9]


7
0
0 1 0 1 0 0 0 0
71
1
4 5
8 9
12 13
55 53 42 43
68 FE AC FF
00 00 00 00
00
72

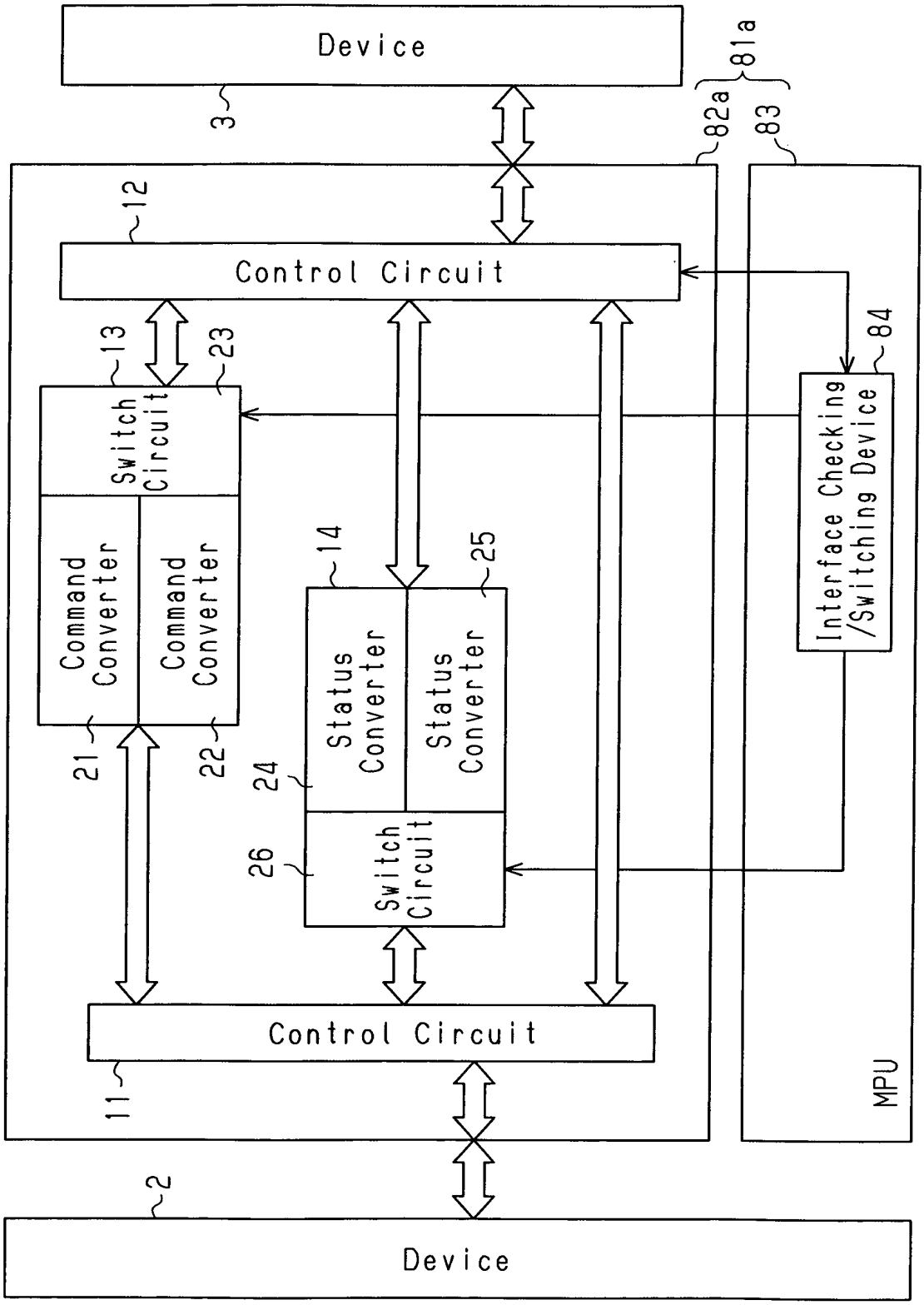
[Fig.10]


7
0
0 1 0 1 0 0 0 0
73
1
4 5
8 9
12 13
55 53 42 43
68 FE AC FF
00 00 00 00
00
74

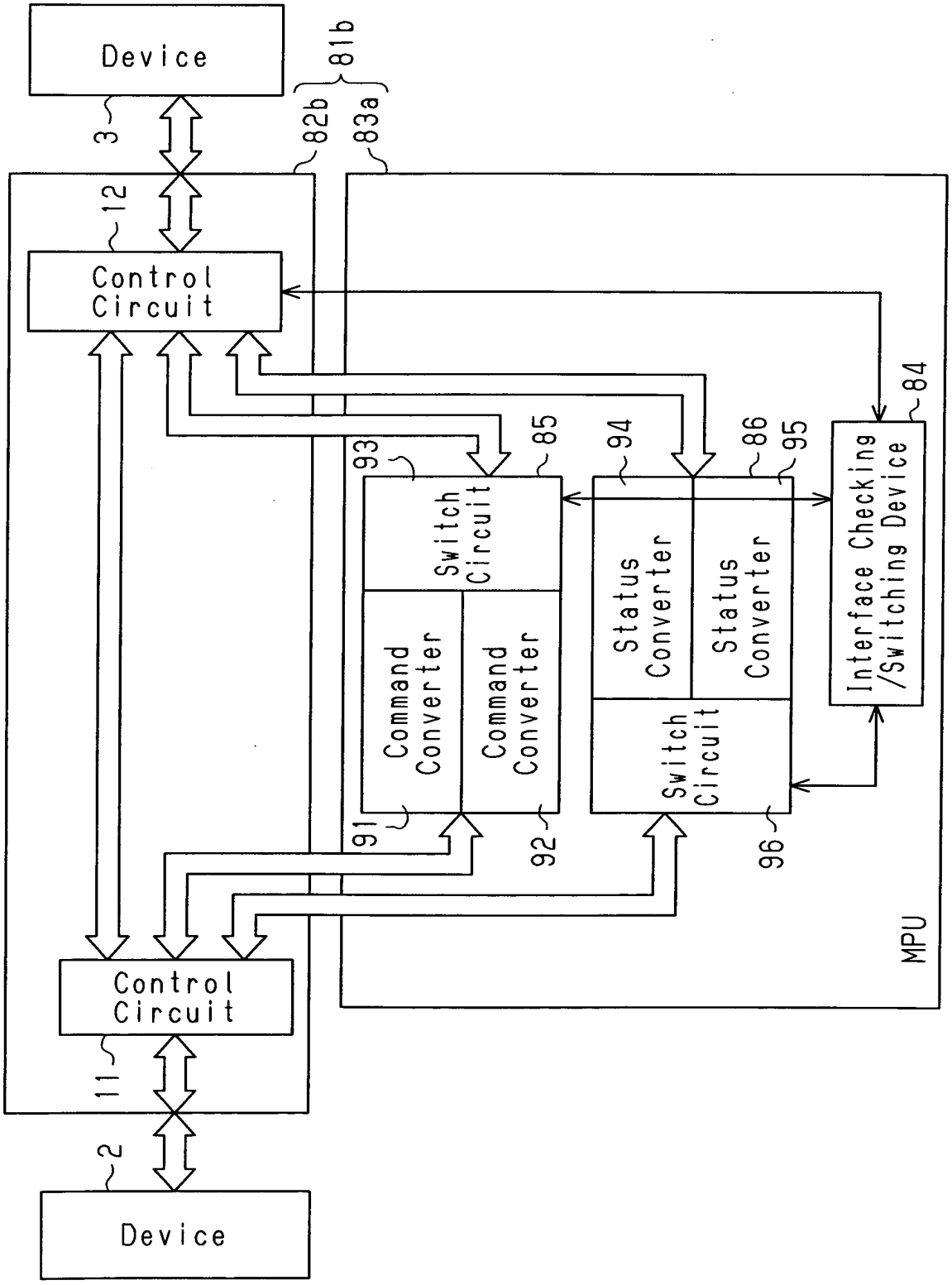
[Fig.11]


Device
3
81
82
83
12
Control Circuit
13
23
84
Switch Circuit
Interface Checking /Switching Device
29
Switch Circuit
Command Converter
Command Converter
14
25
15
Status Converter
Status Converter
Data Converter
Data Converter
21
22
24
27
28
26
Switch Circuit
Control Circuit
11
MPU
2
Device

[Fig.12]


Device
3
81a
82a
83
12
Control Circuit
13
23
84
Switch Circuit
Interface Checking /Switching Device
Command Converter
Command Converter
14
25
Status Converter
Status Converter
21
22
24
26
Switch Circuit
Control Circuit
11
MPU
2
Device

[Fig.13]


Device
3
81b
82b
83a
12
Control Circuit
84
85
94
86
95
93
Switch Circuit
Status Converter
Status Converter
Interface Checking /Switching Device
Command Converter
Command Converter
Switch Circuit
91
92
96
MPU
Control Circuit
11
2
Device